

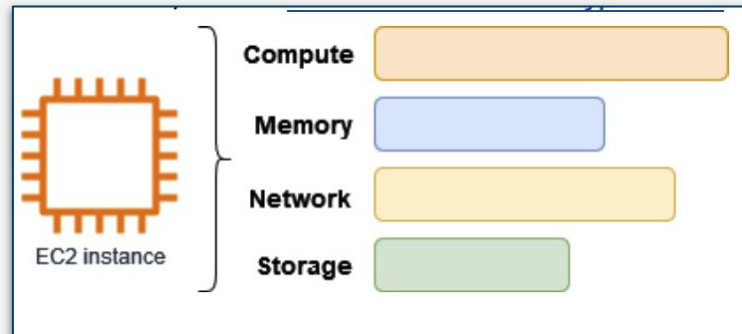
DS 4300

# Amazon EC2 & Lambda

Mark Fontenot, PhD  
Northeastern University

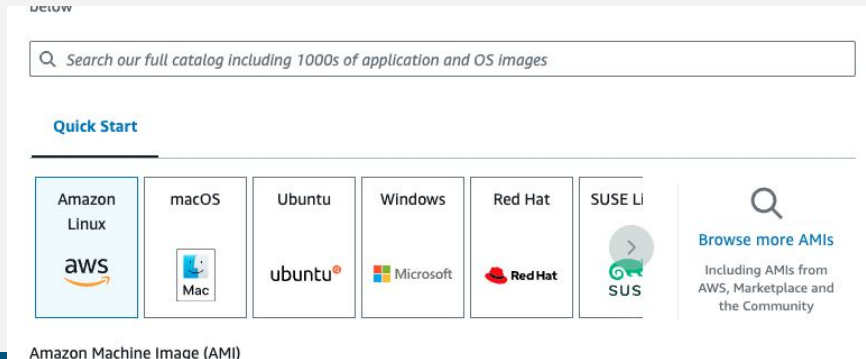
EC2

- EC2 → Elastic Cloud Compute
- Scalable Virtual Computing in the Cloud
- Many (Many!!) instance types available
- Pay-as-you-go model for pricing
- Multiple different Operating Systems



# Features of EC2

- **Elasticity** - easily (and programmatically) scale instances up or down as needed
- You can use one of the standard AMIs OR provide your own AMI if pre-config is needed
- Easily integrates with many other services such as S3, RDS, etc.



AMI = Amazon Machine Image

# EC2 Lifecycle

- **Launch** - when starting an instance for the first time with a chosen configuration
- **Start/Stop** - Temporarily suspend usage without deleting the instance
- **Terminate** - Permanently delete the instance
- **Reboot** - Restart an instance without sling the data on the root volume

# Where Can You Store Data?

- **Instance Store:** Temporary, high-speed storage tied to the instance lifecycle
- **EFS** (Elastic File System) Support - Shared file storage
- **EBS** (Elastic Block Storage) - Persistent block-level storage
- **S3** - large data set storage or EC2 backups even

# Common EC2 Use Cases



- Web Hosting - Run a website/web server and associated apps
- Data Processing - It's a VM... you can do anything to data possible with a programming language.
- Machine Learning - Train models using GPU instances
- Disaster Recovery - Backup critical workloads or infrastructure in the cloud

# Let's Spin Up an EC2 Instance

Search results for 'ec2'

### Services

Show more ▶

-  **EC2** ☆  
Virtual Servers in the Cloud
-  **EC2 Image Builder** ☆  
A managed service to automate build, customize and deploy OS Images

View alarms in CloudWatch (opens new tab)

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Launch instance

Migrate a server

Note: Your instances will launch in the US East (N. Virginia) Region

## Launch an instance

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.


### Name and tags

Name


4300TestServer

Add additional tags


### Quick Start




Amazon Linux




macOS




Ubuntu




Windows



Red Hat



SUSE Linux



Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

### Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type

Free tier eligible

ami-0866a3c8b686eaebea (64-bit (x86)) / ami-0325498274077fac5 (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs



# Let's Spin Up an EC2 Instance

## ▼ Instance type [Info](#) | [Get advice](#)

### Instance type

#### t2.micro

Family: t2 1 vCPU 1 GiB Memory Current generation: true  
On-Demand Windows base pricing: 0.0162 USD per Hour  
On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour  
On-Demand SUSE base pricing: 0.0116 USD per Hour  
On-Demand RHEL base pricing: 0.026 USD per Hour  
On-Demand Linux base pricing: 0.0116 USD per Hour

Free tier eligible

☒ All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

## ▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

Select



Create new key pair

## Create key pair



### Key pair name

Key pairs allow you to connect to your instance securely.

AWS-4300-ec2

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

### Key pair type

☒ RSA

RSA encrypted private and public key pair

☐ ED25519

ED25519 encrypted private and public key pair

### Private key file format

☒ .pem

For use with OpenSSH

☐ .ppk

For use with PuTTY

When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** [Learn more](#)

Cancel

Create key pair

# Let's Spin Up an EC2 Instance

## ▼ Network settings [Info](#)

[Edit](#)

### Network [Info](#)

vpc-025c749a0c68d5aba

### Subnet [Info](#)

No preference (Default subnet in any availability zone)

### Auto-assign public IP [Info](#)

Enable

Additional charges apply when outside of [free tier allowance](#)

### Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group

☐ Select existing security group

We'll create a new security group called 'launch-wizard-1' with the following rules:

☒ Allow SSH traffic from

Helps you connect to your instance

Anywhere

0.0.0.0/0

☐ Allow HTTPS traffic from the internet

To set up an endpoint, for example when creating a web server

☐ Allow HTTP traffic from the internet

To set up an endpoint, for example when creating a web server

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

✕

Cancel

Launch instance

 Preview code

# Ubuntu VM Commands

- Initial user is **ubuntu**
- Access super user commands with **sudo**
- Package manager is **apt**
  - kind of like Homebrew or Choco
- Update the packages installed
  - `sudo apt update`; `sudo apt upgrade`

# MiniConda on EC2

*Make sure you're logged in to your EC2 instance*

- Let's install MiniConda
  - `curl -O https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86\_64.sh`
  - `bash ./Miniconda3-latest-Linux-x86_64.sh`

```
Do you wish to update your shell profile to automatically initialize conda?
This will activate conda on startup and change the command prompt when activated.
If you'd prefer that conda's base environment not be activated on startup,
run the following command when conda is activated:

conda config --set auto_activate_base false

You can undo this by running `conda init --reverse $SHELL`? [yes|no]
[no] >>> yes
```

# Installing & Using Streamlit

- Log out of your EC2 instance and log back in
- Make sure pip is now available:
  - `pip --version`
- Install Streamlit and sklearn
  - `pip install streamlit scikit-learn`
- Make a directory for a small web app
  - `mkdir web`
  - `cd web`

# Basic Streamlit App

```
import streamlit as st

def main():
    st.title("Welcome to my Streamlit App")

    st.write("## Data Sets")
    st.write("""
        - data set 01
        - data set 02
        - data set 03
    """)

    st.write("\n")
    st.write("## Goodbye!")

if __name__ == "__main__":
    main()
```

- nano test.py
- Add code on left
- ctrl-x to save and exit
- streamlit run test.py

# Opening Up The Streamlit Port

Dashboard

EC2 Global View

Events

► Instances

► Images

► Elastic Block Store

▼ Network & Security

Security Groups

Elastic IPs

Placement Groups

Key Pairs

Network Interfaces

☐ - sg-0d58f56d20a97916c launch-wizard-1

**Inbound rules** | Outbound rules | Sharing - new | VPC associations - new | Tags

**Inbound rules (2)** Manage tags Edit inbound rules

<input type="checkbox"/>	Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Desc
<input type="checkbox"/>	-	sg-072f78d9d4ee328fd	IPv6	SSH	TCP	22	::/0	-
<input type="checkbox"/>	-	sg-071d2c2eac0af9f63	IPv4	SSH	TCP	22	0.0.0.0/0	-

## Edit inbound rules

Inbound rules control the incoming traffic that's allowed to reach the instance.

**Inbound rules**

Security group rule ID	Type	Protocol	Port range	Source	Description - optional	
sg-072f78d9d4ee328fd	SSH	TCP	22	Custom	<input type="text" value="::/0"/>	Delete
sg-071d2c2eac0af9f63	SSH	TCP	22	Custom	<input type="text" value="0.0.0.0/0"/>	Delete
-	Custom TCP	TCP	8501	Anywh...	<input type="text" value="0.0.0.0/0"/>	Delete

Cancel

# In a Browser

```
(base) ubuntu@ip-172-31-91-161:~/web$ streamlit run test.py
```

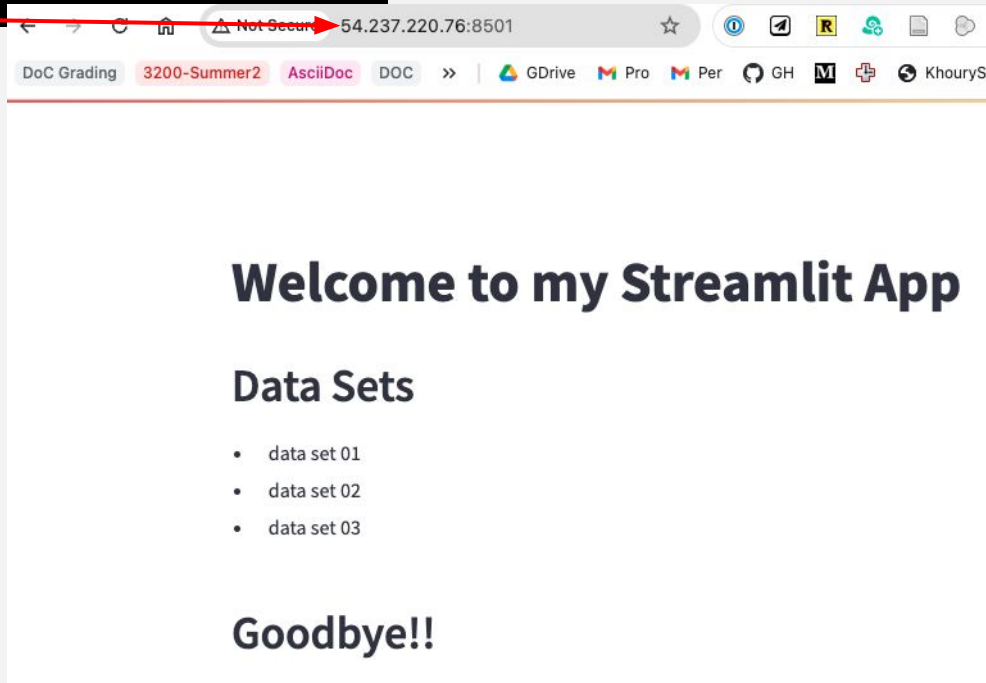
```
Collecting usage statistics. To deactivate, set browser.gatherUsageStats to false.
```

```
You can now view your Streamlit app in your browser.
```

```
Local URL: http://localhost:8501
```

```
Network URL: http://172.31.91.161:8501
```

```
External URL: http://54.237.220.76:8501
```





# AWS Lambda

# Lambdas

- Lambdas provide *serverless computing*
- Automatically run code in response to **events**.
- Relieves you from having to manage servers - only worry about the code
- You only pay for **execution time**, not for idle compute time (different from EC2)

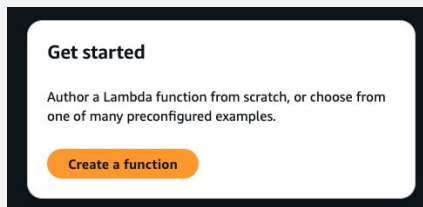
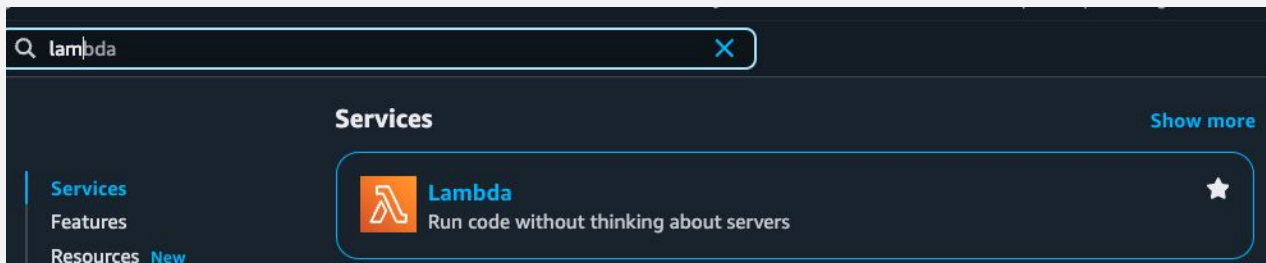
# Lambda Features

- **Event-driven execution** - can be triggered by many different events in AWS
- Supports a large number of runtimes... Python, Java, Node.js, etc
- HIGHLY integrated with other AWS services
- Extremely scalable and can rapidly adjust to demands

## How it Works

- Add/upload your code through AWS MGMT console
- Configure event source(s)
- Watch your Lambda run when one of the event sources fires an event

# Let's Make One



# Making a Lambda

## Create function [Info](#)

Choose one of the following options to create your function.

☐ Author from scratch  
Start with a simple Hello World example.

☒ Use a blueprint  
Build a Lambda application from sample code and configuration presets for common use cases.

☐ Container image  
Select a container image to deploy for your function.

## Basic information [Info](#)

### Blueprint name

Hello world function  
A starter AWS Lambda function.

python3.10 ▲

🔍 Search by blueprint name, runtime or workload category

Learn about Lambda by creating an HTTP endpoint that outputs a server-side rendered web page.

Hello world function  
A starter AWS Lambda function.

nodejs18.x

Hello world function  
A starter AWS Lambda function.

python3.10 ✓

Make an HTTPS request

nodejs18.x

Demonstrates using a built-in Node.js module to make an HTTPS request.

Valid characters are a-z, A-Z, 0-9, hyphens (-), and

# Creating a Function

## Function name

Enter a name that describes the purpose of your function.

SimpleTestFunction

Function name must be 1 to 64 characters, must be uni

## Lambda function code

Code is preconfigured by the chosen blueprint. You can configure it after you create the function. [Learn more](#) about deploying Lambda functions.

① This function contains external libraries.

```
1 import json
2
3 print('Loading function')
4
5
6 def lambda_handler(event, context):
7     #print("Received event: " + json.dumps(event, indent=2))
8     print("value1 = " + event['key1'])
9     print("value2 = " + event['key2'])
10    print("value3 = " + event['key3'])
11    return event['key1'] # Echo back the first key value
12    #raise Exception('Something went wrong')
13
```

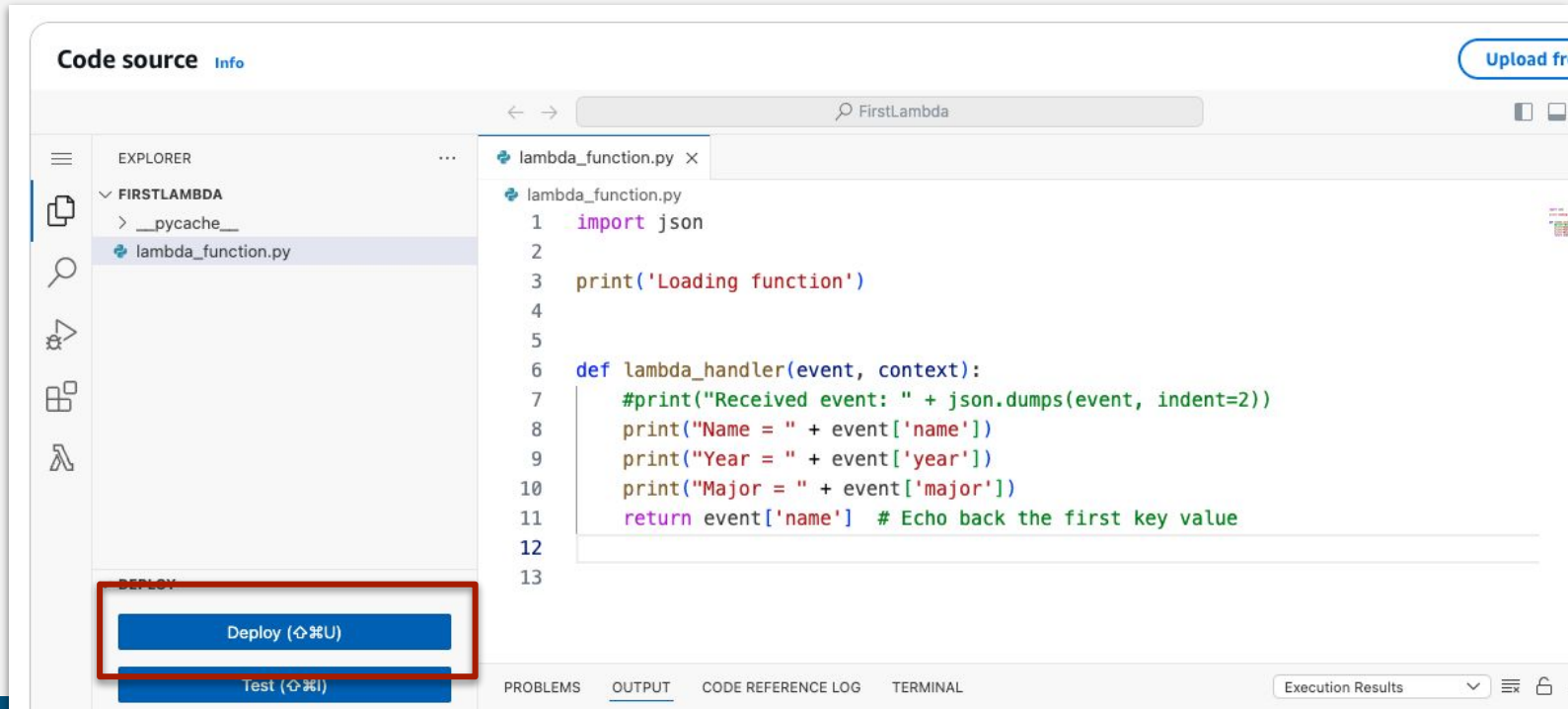
1:1 Python Spaces: 4

Cancel

Create function

# Sample Code

- Edit the code
- Deploy the code!





# Test It

[Code](#) [Test](#) [Monitor](#) [Configuration](#) [Aliases](#) [Versions](#)

**Test event** [Info](#) [CloudWatch Logs Live Tail](#) [Save](#) [Test](#)

To invoke your function without saving an event, configure the JSON event, then choose Test.

**Test event action**  
☒ Create new event ☐ Edit saved event

**Event name**  
  
Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

**Event sharing settings**  
☒ Private  
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. [Learn more](#)  
☐ Shareable  
This event is available to IAM users within the same account who have permissions to access and use shareable events. [Learn more](#)

**Template - optional**

**Event JSON** [Format JSON](#)  

```
1 {  
2   "name": "mark",  
3   "year": "14th",  
4   "major": "CS"  
5 }
```

??